Distributed Systems

COMP90015 2017 Semester 1
Tutorial 08

Questions for today

1. What is a domain transition and how many domain transitions are required for an RPC/RMI call from one process to another on the same host?
2. Compare the worker pool multi-threading architecture with the thread-per-request architecture.
3. What is process migration and explain two circumstances that can prohibit or make it difficult for process migration to take place?
4. Why is shared memory faster than message passing when communicating between processes on the same host? State a reason why shared memory is not preferable when communicating between processes on the same host.

Questions for today

5. What is the difference between a static process location policy and a dynamic process location policy?
6. In the context of load-sharing systems, explain each of the centralized, hierarchical and decentralized designs.
7. What is the difference between sender-initiated load-sharing and receiver-initiated load-sharing? Under what circumstances is sender-initiated more efficient than receiver-initiated?
8. Give two reasons why a monolithic kernel is sometimes more efficient than a micro kernel. Give three reasons why a micro kernel is preferable to a monolithic kernel.

1. What is a domain transition and how many domain transitions are required for an RPC/RMI call from one process to another on the same host?

1. What is a domain transition and how many domain transitions are required for an RPC/RMI call from one process to another on the same host?

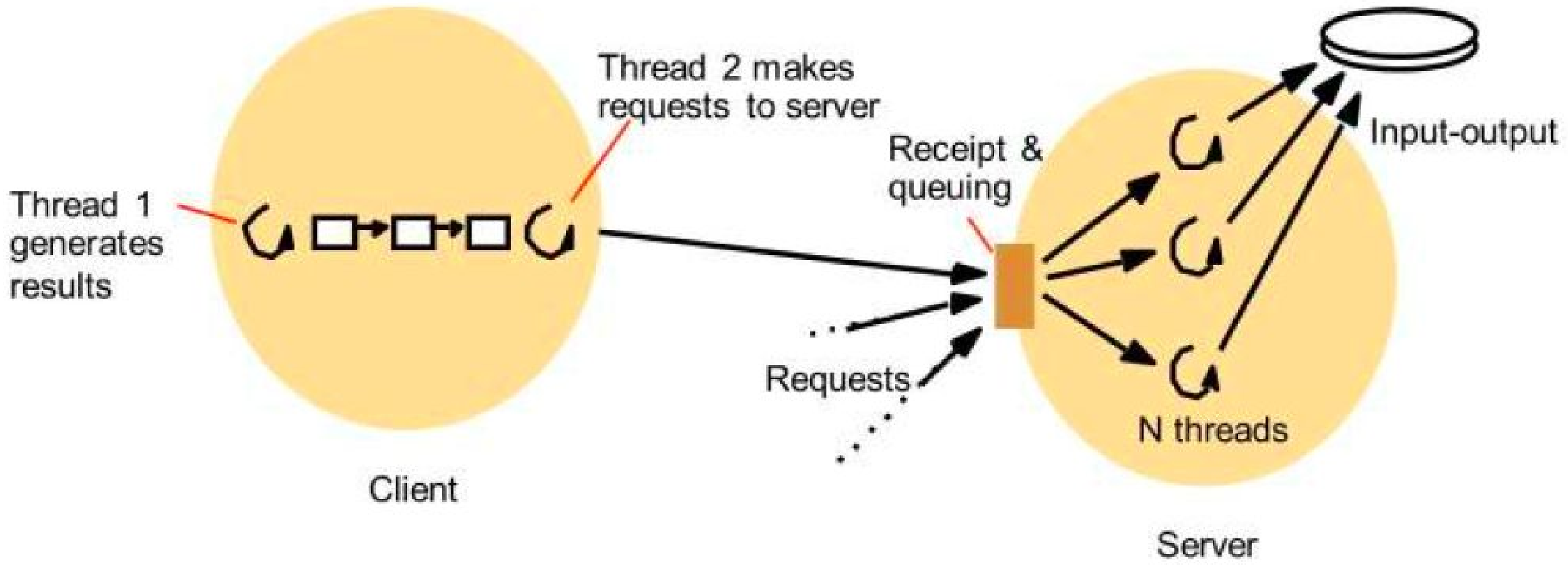
Domain Transition happens when the protection mode is changed, that is, when there is a change from a user context to a kernel context.

As this call is on the same host, the domain transition is to the kernel, to the other process, back to the kernel and back to the calling process.

→ 4 domain transitions.

2. Compare the worker pool multi-threading architecture with the thread-per-request architecture.

Worker pool architecture

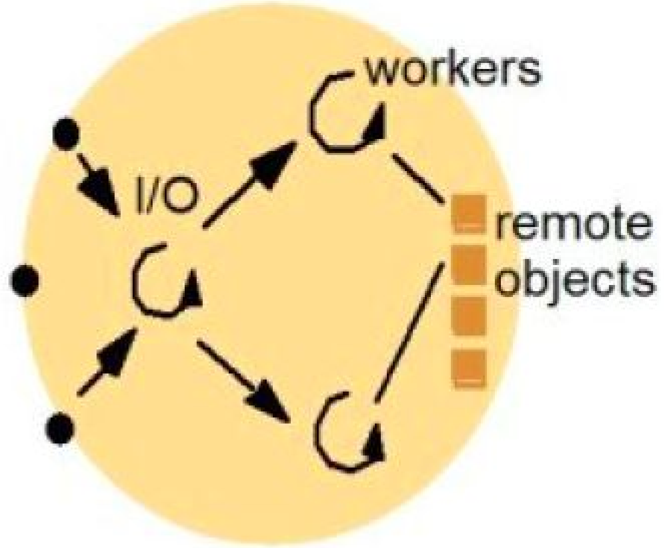


2. Compare the worker pool multi-threading architecture with the thread-per-request architecture.

Worker pool architecture

The server creates a fixed number of threads called a worker pool. As requests arrive at the server, they are put into a queue by the I/O thread and from there assigned to the next available worker thread.

Server creates worker pool ➔ request comes in, put into a queue ➔ assigned to an available worker thread.



a. Thread-per-request

2. Compare the worker pool multi-threading architecture with the thread-per-request architecture.

Thread-per-request architecture

Thread created for each request, when the request is finished, the thread is deallocated.

3. What is process migration and explain two circumstances that can prohibit or make it difficult for process migration to take place?

3. What is process migration and explain two circumstances that can prohibit or make it difficult for process migration to take place?

Process Migration is a form of process management where the process is moved from one host to another.

This can be done by copying their address space.

It can be difficult if:

- The process code is CPU dependent.
- The process is using host resources such as open files and sockets.

4. Why is shared memory faster than message passing when communicating between processes on the same host? State a reason why shared memory is not preferable when communicating between processes on the same host.

4. Why is shared memory faster than message passing when communicating between processes on the same host? State a reason why shared memory is not preferable when communicating between processes on the same host.

When we use message passing, there is an overhead of marshaling, transmission and unmarshalling the messages. There is no such overhead in a shared memory.

Disadvantage:

When using shared memory processes are not protected from each other, if we want more security and protection we should use message passing.

5. What is the difference between a static process location policy and a dynamic process location policy?

5. What is the difference between a static process location policy and a dynamic process location policy?

Location Policy

Determines which host, from a set of given hosts, the new process should be allocated on.

5. What is the difference between a static process location policy and a dynamic process location policy?

Static Policies

Operates without regard to the current state of the distributed system.

Adaptive Policies

Receives feedback about the current state of the distributed system and makes a decision based on that. For example, load balancing.

6. In the context of load-sharing systems, explain each of the centralized, hierarchical and decentralized designs.

6. In the context of load-sharing systems, explain each of the centralized, hierarchical and decentralized designs.

Load managers collect information about the nodes and use it to allocate new processes to nodes.

6. In the context of load-sharing systems, explain each of the centralized, hierarchical and decentralized designs.

Centralized

There is 1 load manager that receives feedback from all the hosts in the system.

Hierarchical

There are several load managers, organized in a tree structure. The hosts are the leaf nodes.

Decentralized

Nodes exchange information with one another directly to make allocation decisions.

7. What is the difference between sender-initiated load-sharing and receiver-initiated load-sharing? Under what circumstances is sender-initiated more efficient than receiver-initiated?

7. What is the difference between sender-initiated load-sharing and receiver-initiated load-sharing? Under what circumstances is sender-initiated more efficient than receiver-initiated?

Transfer Policy

Determines whether the new process is allocated locally or remotely.

7. What is the difference between sender-initiated load-sharing and receiver-initiated load-sharing? Under what circumstances is sender-initiated more efficient than receiver-initiated?

Sender-initiated (push)

The node that requires a new process to be created is responsible for initiating the transfer decision.

The transfer is usually initiated when its own load crosses a threshold.

Receiver-initiated (pull)

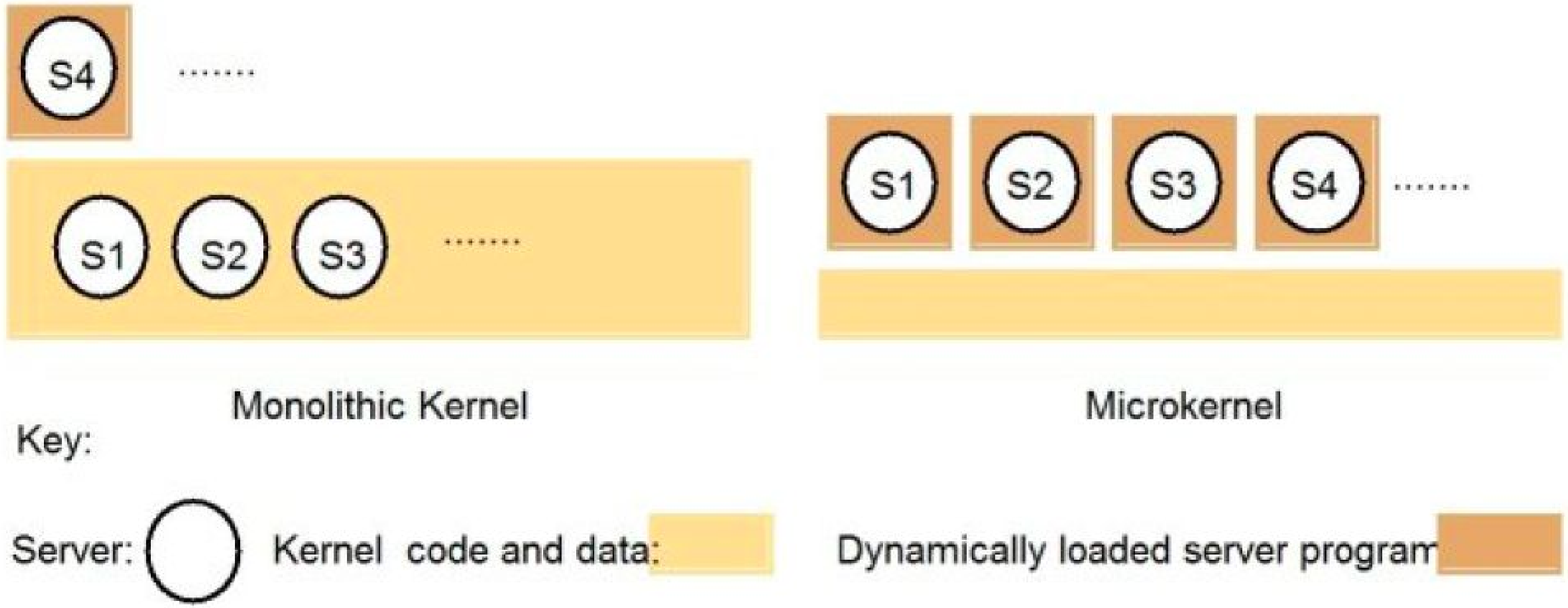
A node whose load is below a given threshold will advertise its existence to other nodes so that the relatively loaded nodes can transfer work to it.

7. What is the difference between sender-initiated load-sharing and receiver-initiated load-sharing? Under what circumstances is sender-initiated more efficient than receiver-initiated?

When to use sender and when to use receiver?

- Sender-initiated policies are preferable to receiver-initiated policies at light to moderate system loads.
- Receiver-initiated policies are preferable at high system loads, but only if the costs of task transfer under the two strategies are comparable.
- If the cost of task transfers under receiver-initiated policies is significantly greater than under sender-initiated policies (for example, because executing tasks must be transferred), then sender-initiated policies provide uniformly better performance.

8. Give two reasons why a monolithic kernel is sometimes more efficient than a micro kernel. Give three reasons why a micro kernel is preferable to a monolithic kernel.


S4
....
S1
S2
S3
....
Monolithic Kernel
S1
S2
S3
S4
....
Microkernel
Key:
Server:
Kernel code and data:
Dynamically loaded server program

8. Give two reasons why a monolithic kernel is sometimes more efficient than a micro kernel. Give three reasons why a micro kernel is preferable to a monolithic kernel.

Monolithic Kernel provides diverse functionality.

Complex functionality can be provided through a smaller number of system calls and interprocess communication is minimized.

Micro Kernel provides only fundamental functionality.

Allow a greater extensibility and have a greater ability to enforce modularity behind memory protection boundaries.

A small kernel is also more likely to be free of bugs.